Constrained minimization of Manhattan distance for voting on budgets

Jan Behrens

2024-11-04 10:00 UTC

1 Problem

We have a certain budget or resource, e. g. 1000 person hours or \$ 1,000,000 to be assigned to d candidates (e.g. projects). We assume n voters, which each propose a relative budget allocation for each candidate, e.g. $(0.6, 0.3, 0.1)$ if he/she/they wants to assign 60 % to the first, 30 % to the second, and 10 % of the budget to the third candidate. In the following, $X \in \mathbb{R}^{n \times d}$ is a $n \times d$ matrix containing all candidates as columns and all voters as rows. X_{ij} is voter i's desired relative budget for candidate j.

There are certain constraints on the voter's ballots. Equations (1) below ensure that no voter can spend less than 0 % or more than 100 % on a particular candidate on their respective ballot, and equations (2) ensure that each voter must use 100 % of the total budget[1].

$$\forall i \, \forall j : 0 \leq X_{ij} \leq 1 \tag{1}$$

$$\forall i : \sum_{j=1}^{d} X_{ij} = 1 \tag{2}$$

We search a suitable voting algorithm $f : X \mapsto w$, which maps all votes X to a result w that assigns each candidate j a certain budget $w_j \in [0, 1]$. For the output w of the voting algorithm, we demand $\sum w_j = 1$ such that the total available budget is spent, but not more:

$$f : \left\{ X \in [0,1]^{n \times d} \mid \forall i : \sum_{j=1}^{d} X_{ij} = 1 \right\} \rightarrow \left\{ x \in [0,1]^{d} \mid \sum_{j=1}^{d} x_j = 1 \right\} \tag{3}$$

$$f : X \mapsto w \tag{4}$$

Moreover, the algorithm f shall be designed in such a way that each voter has little incentive for tactical voting.

[1] Note that in cases where there should be an option to not spend all resources, it's always possible to add another candidate labeled "save the money/resource", if desired.

2 Proposed algorithm

Let $\|x\|_1$ denote the Manhattan norm and $\|x\|_2$ the Euclidean norm of x, i.e.:

$$\|x\|_1 = \sum_j |x_j| \tag{5}$$

$$\|x\|_2 = \sqrt{\sum_j x_j^2} \tag{6}$$

We first calculate the geometric median g:

$$g = \operatorname*{arg\,min}_{x \in \mathbb{R}^d} \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{d} (x_j - X_{ij})^2} \tag{7}$$

$$= \operatorname*{arg\,min}_{x \in \mathbb{R}^d} \sum_{i=1}^{n} \|x - X_i\|_2 \tag{8}$$

Note that the geometric median may be undefined in the rare case of n being odd *and* all votes being on a 1-dimensional line. In this case, we could use the 1-dimensional median and take the arithmetic mean of the two middle values.

Now the assigned budgets w_j are:

$$w = \operatorname*{arg\,min}_{x \in \mathbb{R}^d, \sum_j x_j = 1} \left[\frac{\sum_{j=1}^{d} |x_j - g_j|}{2} + \sum_{i=1}^{n} \sum_{j=1}^{d} |x_j - X_{ij}| \right] \tag{9}$$

$$= \operatorname*{arg\,min}_{x \in \mathbb{R}^d, \sum_j x_j = 1} \left[\frac{\|x - g\|_1}{2} + \sum_{i=1}^{n} \|x - X_i\|_1 \right] \tag{10}$$

Here, $\sum x_j = 1$ ensures that not more and not less than the total available budget is used and $\frac{\|x-g\|_1}{2}$ serves as a tie-breaker, which corresponds to adding half a vote for the geometric median.

3 Reasoning

Disregarding the tie-breaking and the constraint that the budget must be spent and not be exceeded, the definition of w corresponds to minimizing the sum of Manhattan distances between w and the votes X_i. This can be seen as an equillibrium process where each voter tries to increase the budgets for those candidates where they desire a higher budget and to reduce the budgets for those candidates where they desire a lower budget, and where each voter can transfer the same amount at the same time but not increase or reduce the total budget spent (i.e. each increase must have an accompanying reduction). Because it doesn't matter how much higher or lower a voter's wish for a particular candidate's budget is in this process, there is little incentive for tactical voting. I. e. there is little incentive to express polarized opinions where a voter, for example, assigns all resources to one candidate in the attempt to increase that candidate's budget further.

4 Examples

The following examples E_k have been calculated with the computer program given in the next section.

$$E_1 = \begin{pmatrix} 100\,\% & 0\,\% \\ 58\,\% & 42\,\% \\ 0\,\% & 100\,\% \end{pmatrix} \tag{11}$$

$$f(E_1) = \begin{pmatrix} 58\,\% & 42\,\% \end{pmatrix}^\mathsf{T} = g \tag{12}$$

$$E_2 = \begin{pmatrix} 60\,\% & 30\,\% & 10\,\% \\ 20\,\% & 60\,\% & 20\,\% \\ 25\,\% & 25\,\% & 50\,\% \end{pmatrix} \tag{13}$$

$$f(E_2) \approx \begin{pmatrix} 33.11\,\% & 39.60\,\% & 27.28\,\% \end{pmatrix}^\mathsf{T} = g \tag{14}$$

$$E_3 = \begin{pmatrix} 60\,\% & 30\,\% & 10\,\% \\ 20\,\% & 60\,\% & 20\,\% \\ 0\,\% & 0\,\% & 100\,\% \end{pmatrix} \tag{15}$$

$$f(E_3) \approx \begin{pmatrix} 30.28\,\% & 42.39\,\% & 27.34\,\% \end{pmatrix}^\mathsf{T} = g \tag{16}$$

$$E_4 = \begin{pmatrix} 60\,\% & 30\,\% & 10\,\% \\ 60\,\% & 30\,\% & 10\,\% \\ 60\,\% & 30\,\% & 10\,\% \\ 60\,\% & 30\,\% & 10\,\% \\ 60\,\% & 30\,\% & 10\,\% \\ 20\,\% & 60\,\% & 20\,\% \\ 20\,\% & 60\,\% & 20\,\% \\ 20\,\% & 60\,\% & 20\,\% \\ 0\,\% & 0\,\% & 100\,\% \\ 0\,\% & 0\,\% & 100\,\% \\ 0\,\% & 0\,\% & 100\,\% \\ 0\,\% & 0\,\% & 100\,\% \end{pmatrix} \tag{17}$$

$$f(E_3) = \begin{pmatrix} 50\,\% & 30\,\% & 20\,\% \end{pmatrix}^\mathsf{T} \neq g \tag{18}$$

In examples E_1 through E_3, tie-breaking is used and the results are also equivalent to the geometric median g. Example E_4 shows a case where tie-breaking is not needed and the geometric median has no influence on the result.

5 Implementation in Octave

```
function w = budget_voting(X)
  # X is matrix with voters as rows and candidates as columns;
  # each entry reflects a budget assigned by the voter to the candidate.

  # Xt is X transposed, i.e. candidates as rows and voters as columns.
  Xt = transpose(X);

  # Xtn is Xn normalized such that each voter assigns a total budget of 1.
  Xtn = Xt ./ sum(Xt, 1);

  # eudists(x) contains the Euclidean distances between x and each vote.
  eudists = @(x) sqrt(sum((x-Xtn).^2, 1));

  # eudist(x) is the sum of all Euclidean distances between x and each vote.
  eudist = @(x) sum(eudists(x), 2);

  # eudist_grad(x) is the gradient of eudist(x) as a column vector,
  # used to aid optimization.
  eudist_grad = @(x) sum((x-Xtn) ./ eudists(x), 2);

  # The geometric median minimizes the sum of all Euclidean distances:
  g = sqp(mean(Xtn, 2), {eudist, eudist_grad});

  # mandist(x) contains the Manhattan distances between x and each vote
  # plus a tie-breaker (weighted 1/2) in favor of the geometric median.
  mandist = @(x) sum(sum(abs(x-Xtn), 1), 2) + sum(abs(x-g), 1)/2;

  # mandist_grad(x) is the gradient of mandist(x) as a column vector,
  # used to aid optimization.
  mandist_grad = @(x) sum(sign(x-Xtn), 2) + sign(x-g)/2;

  # excess(x) is the sum of the candidates' budgets minus 1.
  excess = @(x) sum(x, 1) - 1;

  # excess_grad_t(x) is the gradient of excess(x) as a row vector,
  # used to aid optimization.
  excess_grad_t = @(x) ones(1, rows(x));

  # The solution minimizes the Manhattan distance (including the tie-breaker)
  # under the condition that the sum of the candidates' budgets is 1.
  w = sqp(g, {mandist, mandist_grad}, {excess, excess_grad_t});
end
```

Note that this implementation may fail in the rare case of the geometric median being undefined (even number of voters and all votes on one line).